Matrix Ordering of 3D Point Clouds for Performance Improvement in Neighbor Searching Through Bounding

Abel Rodríguez Calleja
Francisco Fernández Rivera
Miguel Yermo García
Tomás García Barreiro

Universidade de Santiago de Compostela
Departamento de Electrónica e Computación
Centro Singular de Investigación en Tecnoloxías Intelixentes (CiTIUS)





Table of contents I

1. Introduction
   - Point clouds
   - Neighbor searching
   - Motivation
   - Approaches for neighbor searching
   - How is the search performed?
2. Materials
   - Test environment
   - Dataset
3. Experiments

Table of contents II

4 Results
- Ordering
- Creation of data structures
- Neighbors searching

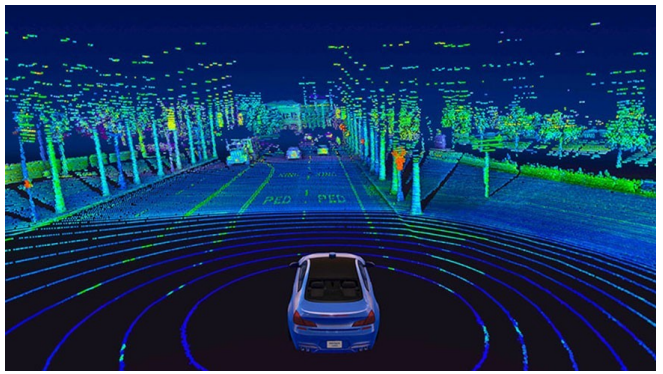
5 Conclusions and potential expansions

What is a point cloud?

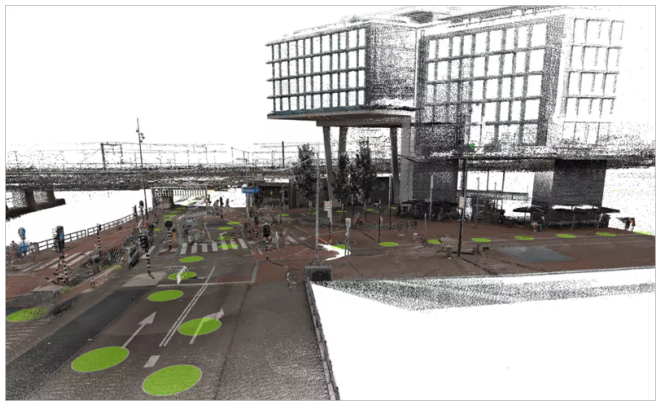


What is neighbor searching and why is it important?

- Filtering
- Segmentation
- Normal estimation
- Geometric reconstruction
- ...

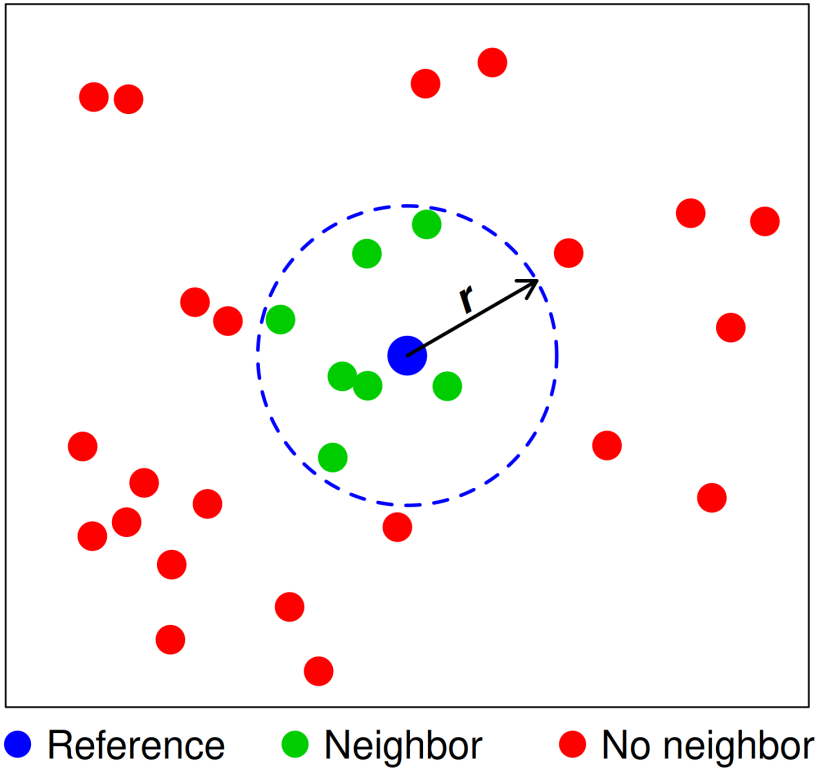


Automotive

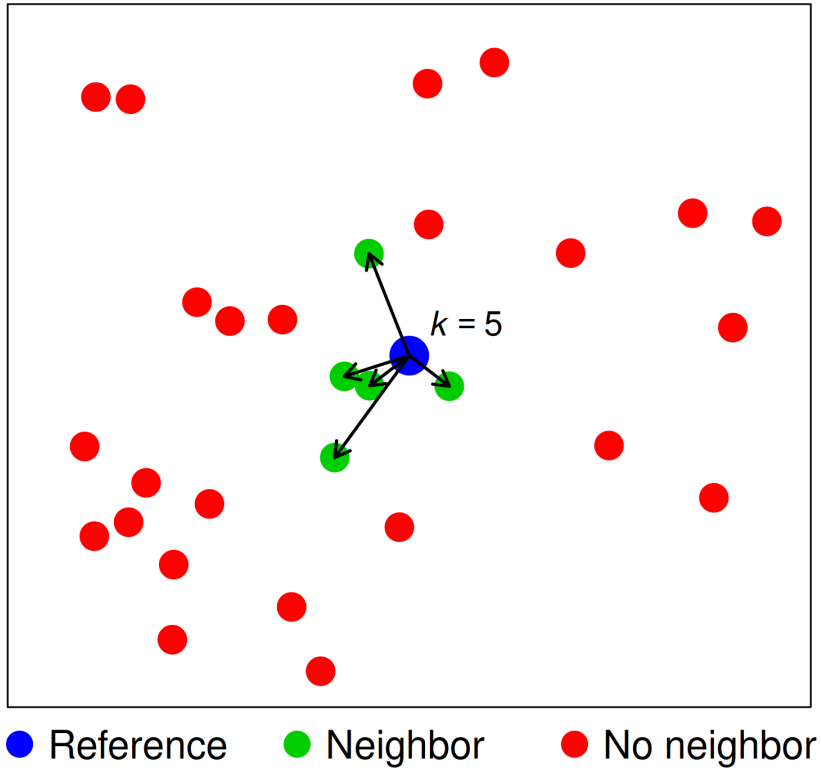


Architecture

Types of neighbor searches

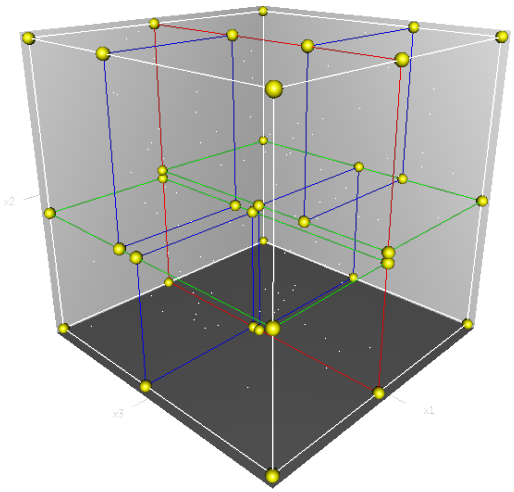


Radius neighbor search.

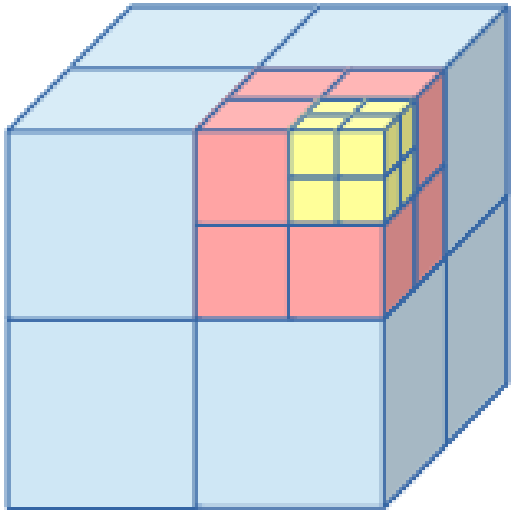


K-nn neighbor search.

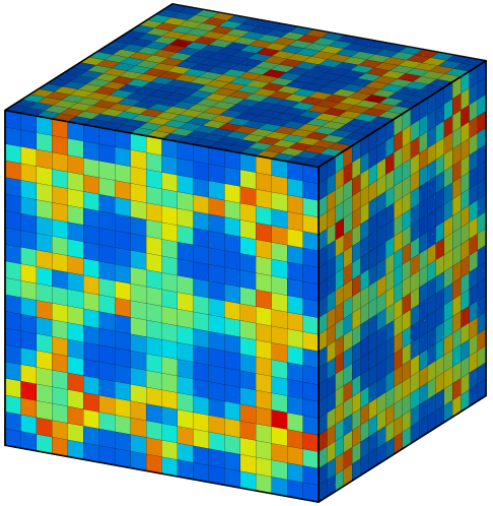
Traditional methods



K-d tree

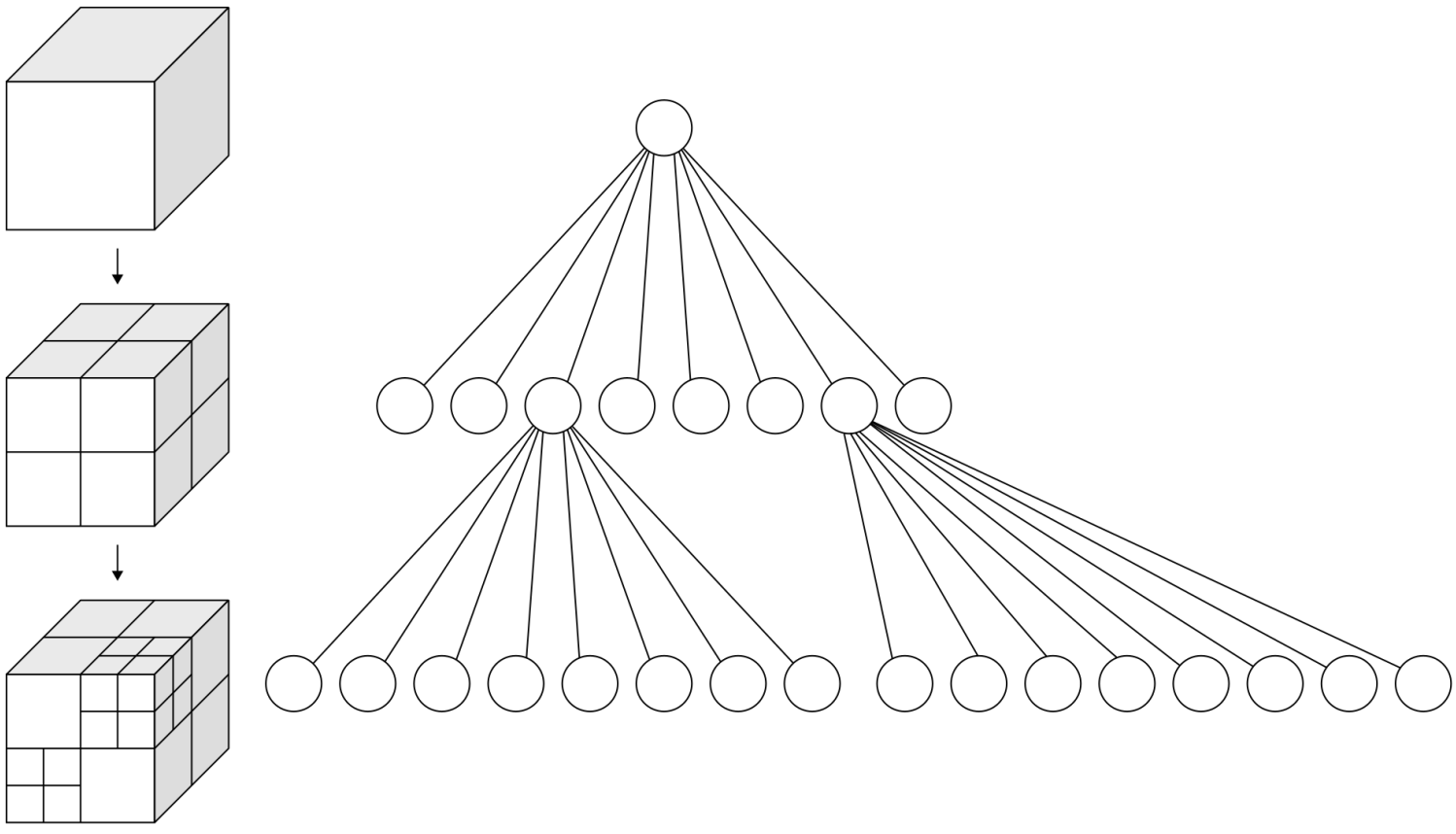


Octree



Voxel grid

Point-region octree

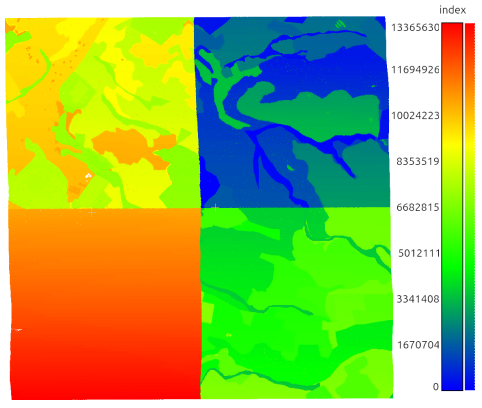


Recursive subdivision of a cube into octants and its corresponding octree.

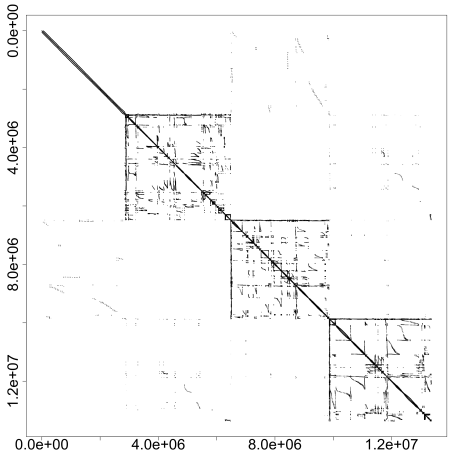
Proposal of this work

1. **Calculate the neighborhood matrix** of a point cloud using a sorting radius.
2. **Order the matrix** using the *Reverse Cuthill-McKee* algorithm.
3. **Apply the permutation** obtained to the point cloud.
4. Perform **neighbor searches using bounding** on the ordered point cloud.

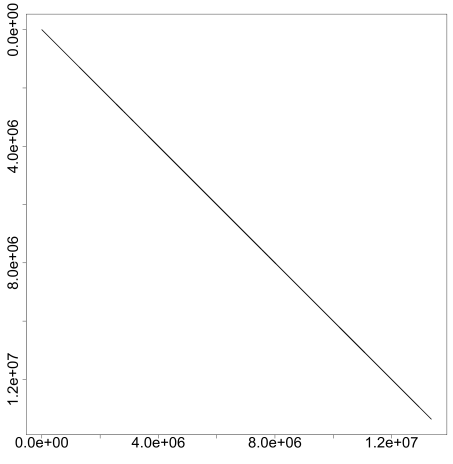
Steps



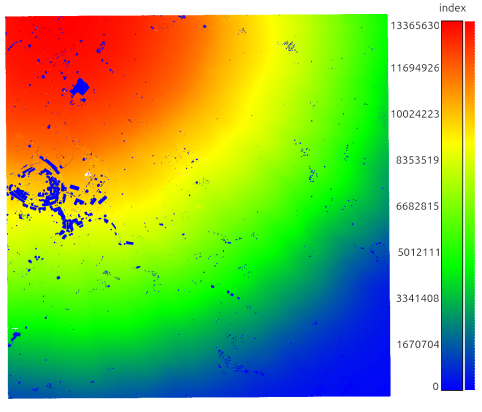
Calculate neighborhood matrix



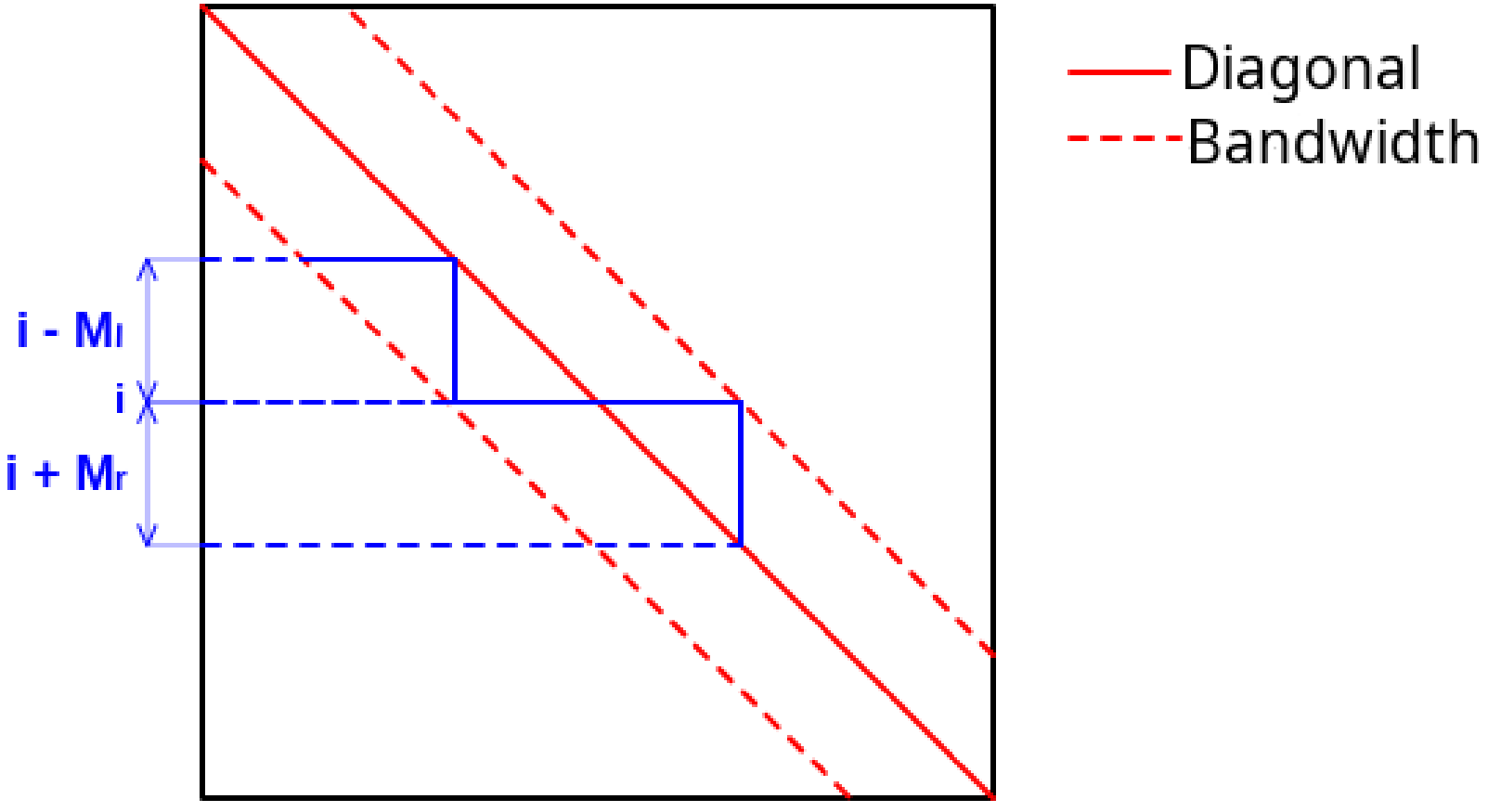
Order with RCM



Apply ordering



Search radius = sorting radius



Search for $r_{search} = r_{order}$.

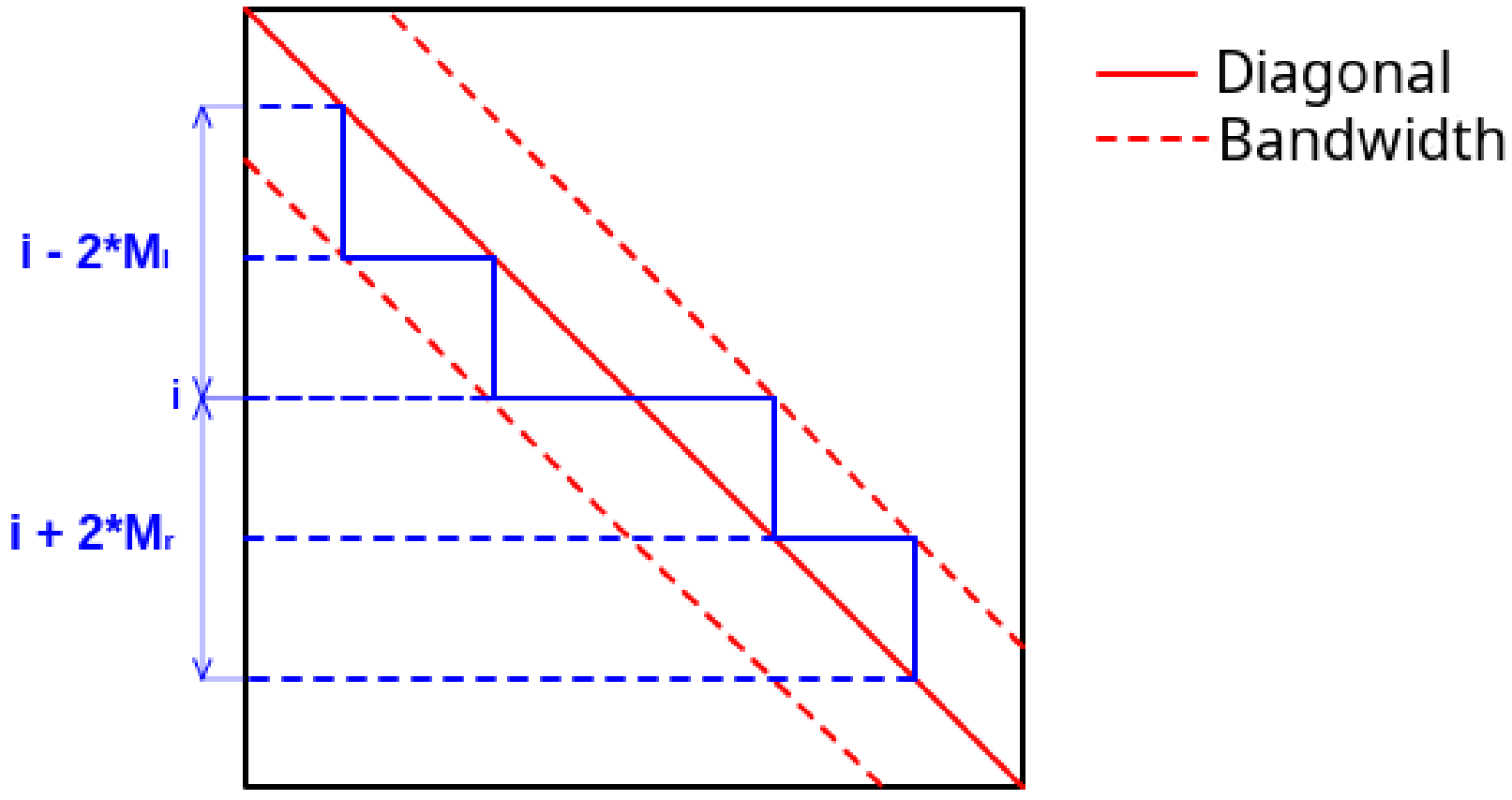
Search radius $<$ sorting radius

For $r_{search} < r_{order}$:

$$\mathcal{N}_{r_{search}}(i) \subseteq \mathcal{N}_{r_{order}}(i)$$

Where $\mathcal{N}_r(i)$ is the neighborhood function that, given a radius r and a point i, returns the set of neighbors of i within that radius.

Search radius > sorting radius



Search for $r_{search} > r_{order}$ $\left(\left\lceil \frac{r_{search}}{r_{order}} \right\rceil = 2\right)$.

FinisTerrae III



Component	**Specification**
Processors	2 x Intel Xeon Ice Lake 8352Y, 32 cores at 2.2 GHz
Memory	256 GiB
Storage	960 GiB SSD NVMe
Operating system	Rocky Linux 8.4
Compilers	GCC and G++ 14.3.0
Modules	cesga/2025 gcc/14.3.0

Point clouds

Dataset	Cloud	Type of capture	# of points
PNOA-LiDAR	PNOA	Aerial	13.365.631
DALES	5095_54440	Aerial	11.703.345
	5085_54320	Aerial	14.394.749
Hessigheim 3D	Mar18_test	Aerial (UAV)	51.744.758
	Mar18_train	Aerial (UAV)	59.445.106
Paris-Lille-3D	Paris	Terrestrial (vehicle)	50.000.000
	Lille	Terrestrial (vehicle)	80.000.000
Semantic3D	Bildstein_station1	Terrestrial (tripod)	29.697.591
	Station1	Terrestrial (tripod)	50.122.464
	Sg27_station8	Terrestrial (tripod)	429.615.314
Speulderbos	Speulderbos	Terrestrial	721.810.646

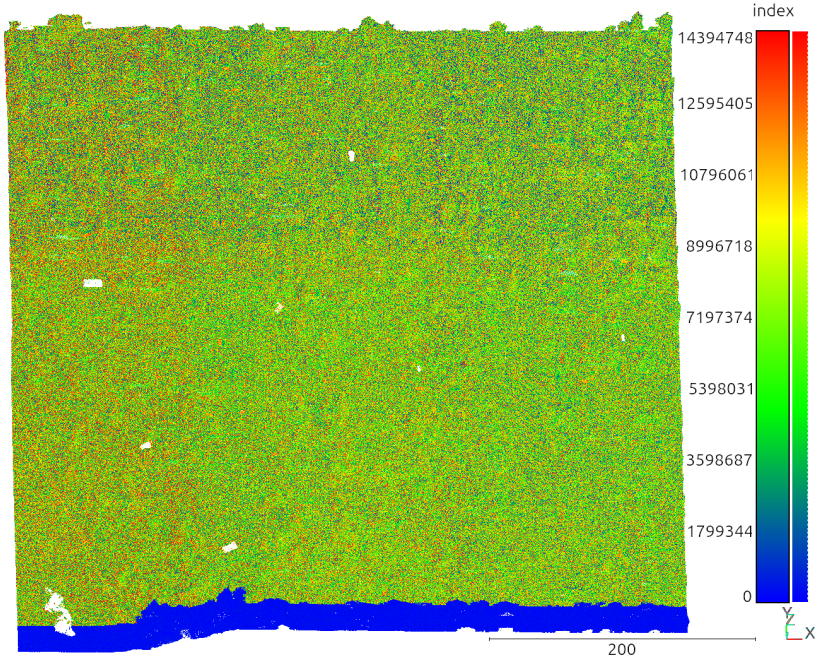
Experiments

- Point cloud ordering
- Structure creation
- Neighbor search

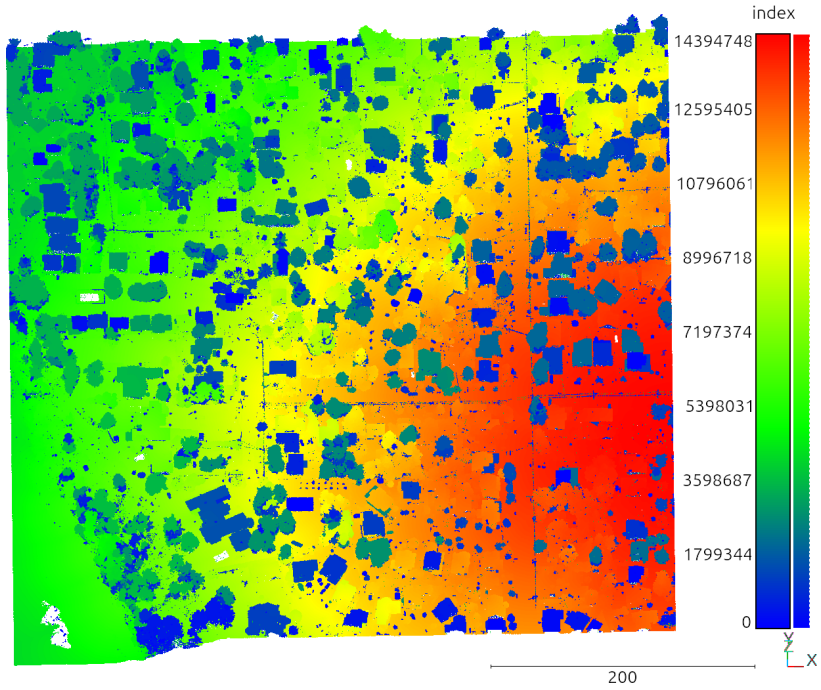
Search radii

Radius	**Avg. neighbors per point**
Small	< 10
Medium	$10 - 40$
Large	> 70

Qualitative analysis - Cloud 5085_54320



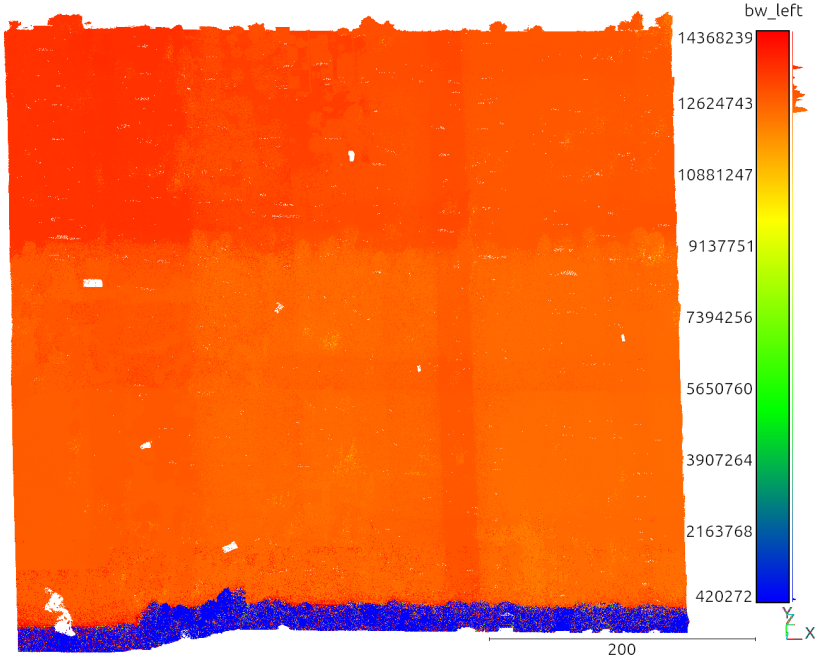
Unsorted



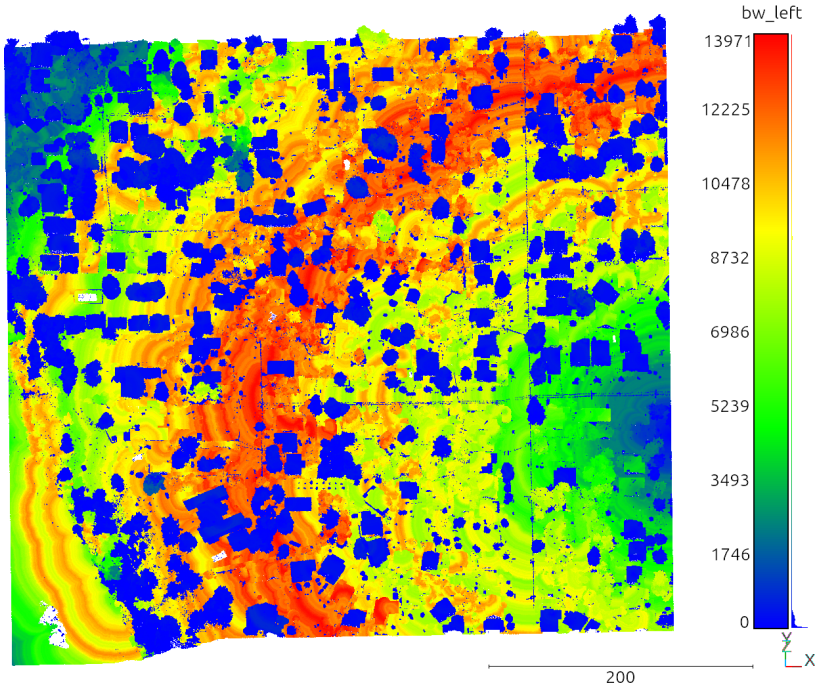
Ordered for $r = 0{,}5$

Clouds colored based on the **indices** of the points.

Qualitative analysis - Cloud 5085_54320



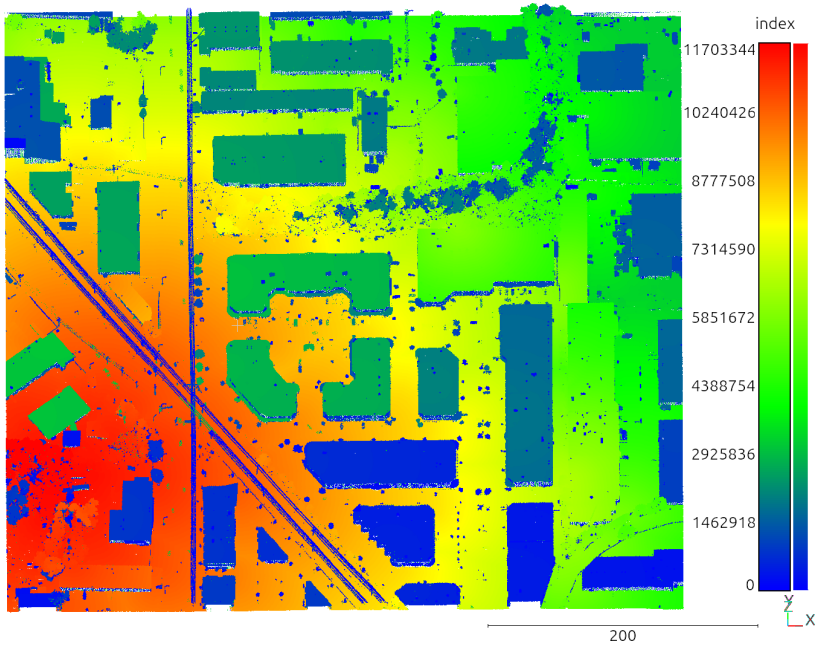
Unsorted



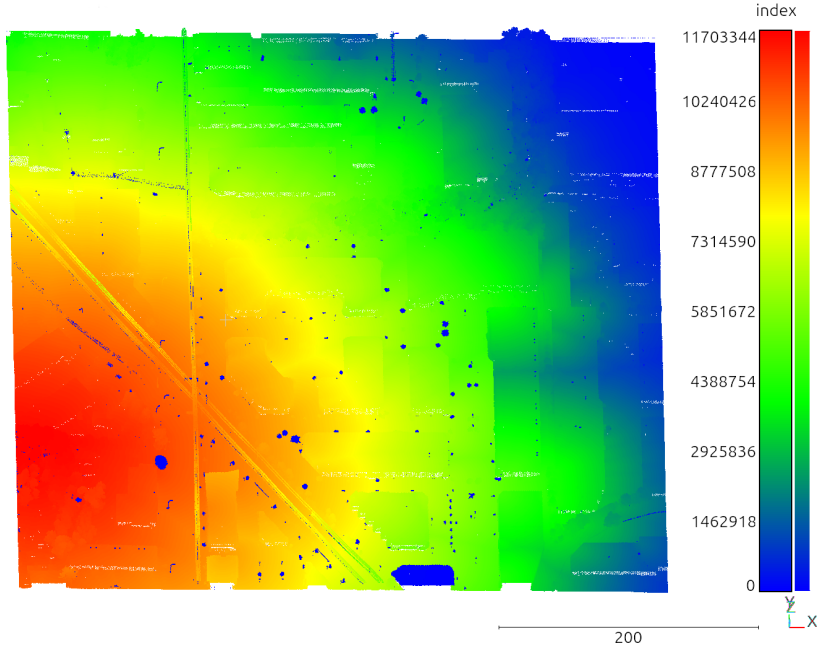
Ordered for $r = 0{,}5$

Clouds colored based on the **bandwidth** of the points.

Qualitative analysis - Cloud 5095_54440



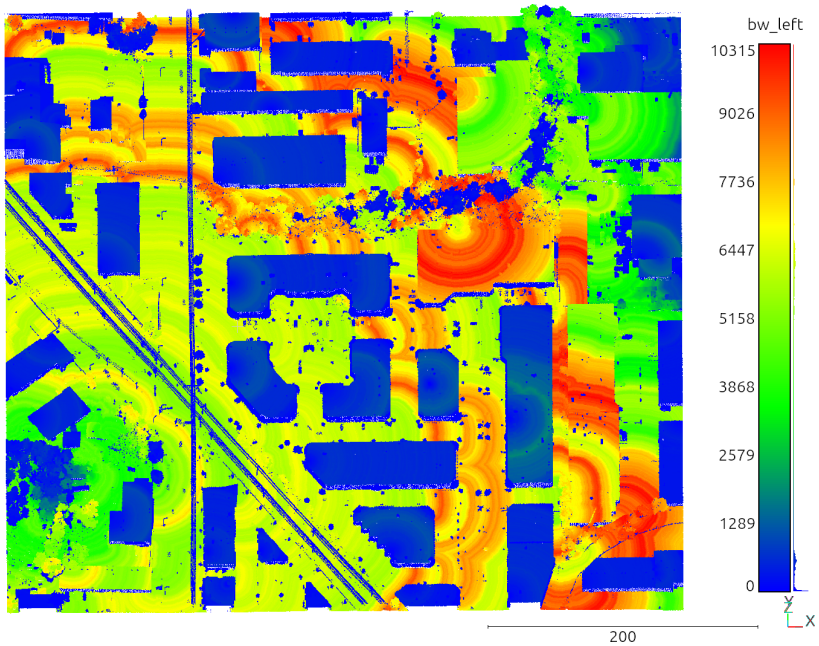
Ordered for $r = 0{,}5$



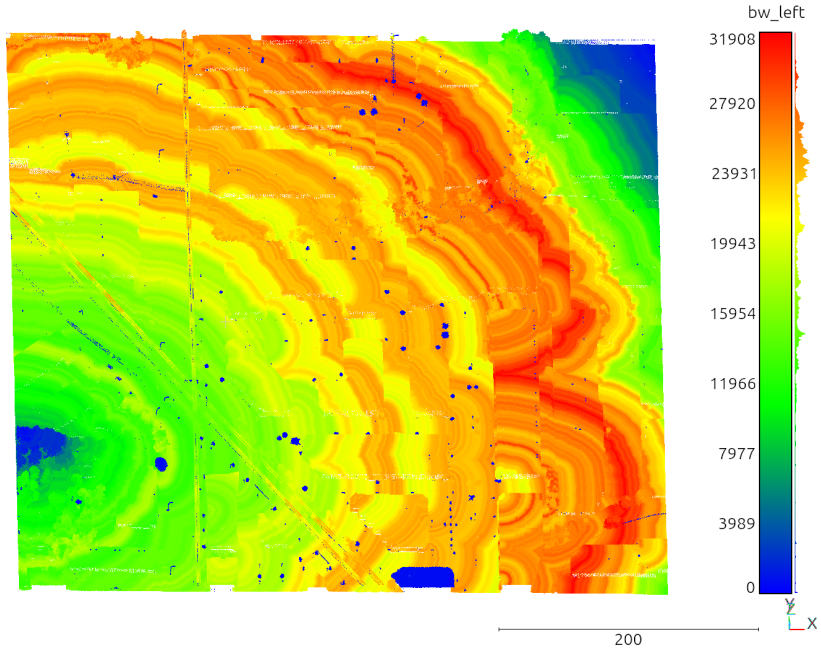
Ordered for $r = 1$

Clouds colored based on the **indices** of the points.

Qualitative analysis - Cloud 5095_54440



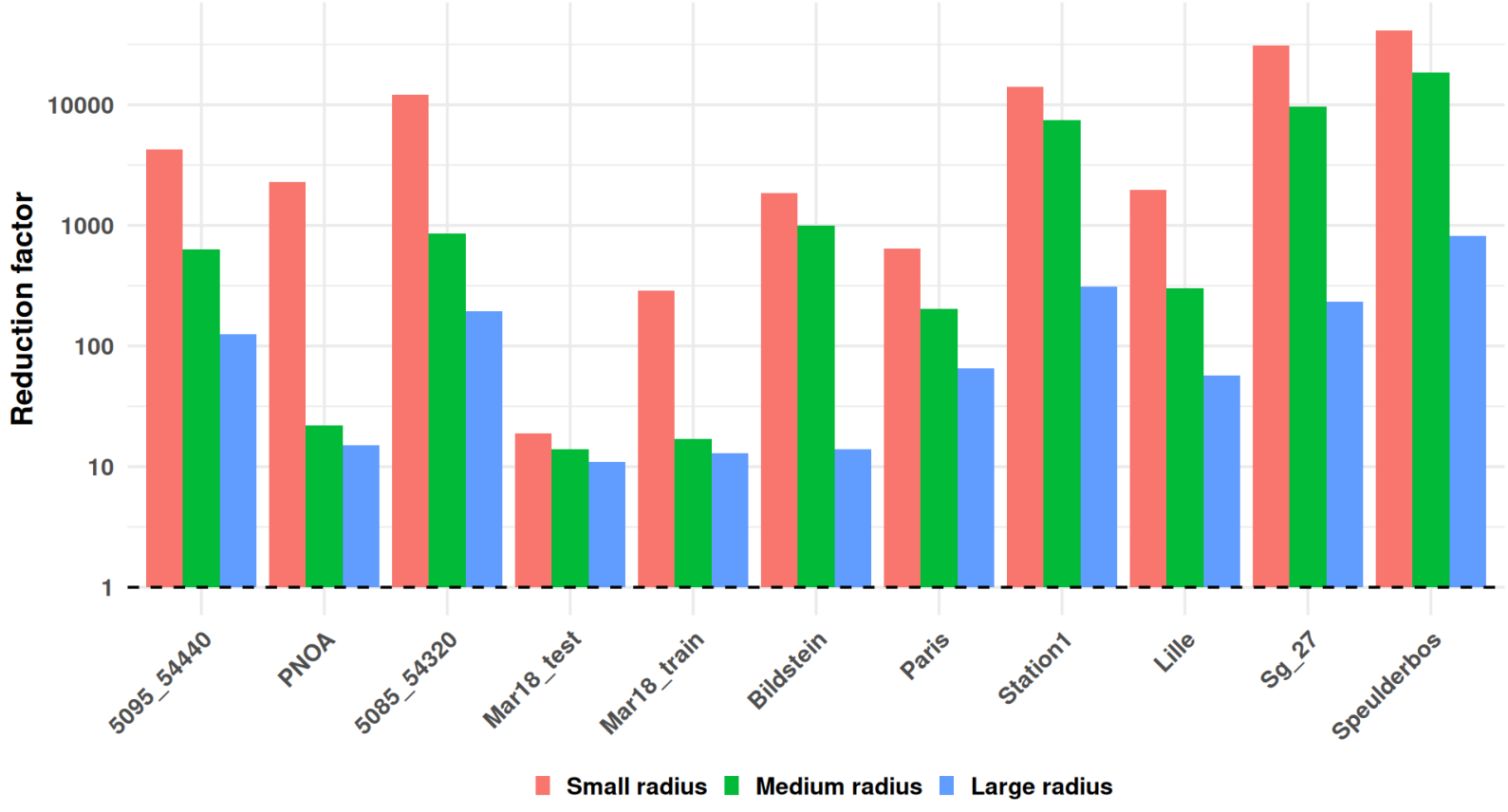
Ordered for $r = 0{,}5$



Ordered for $r = 1$

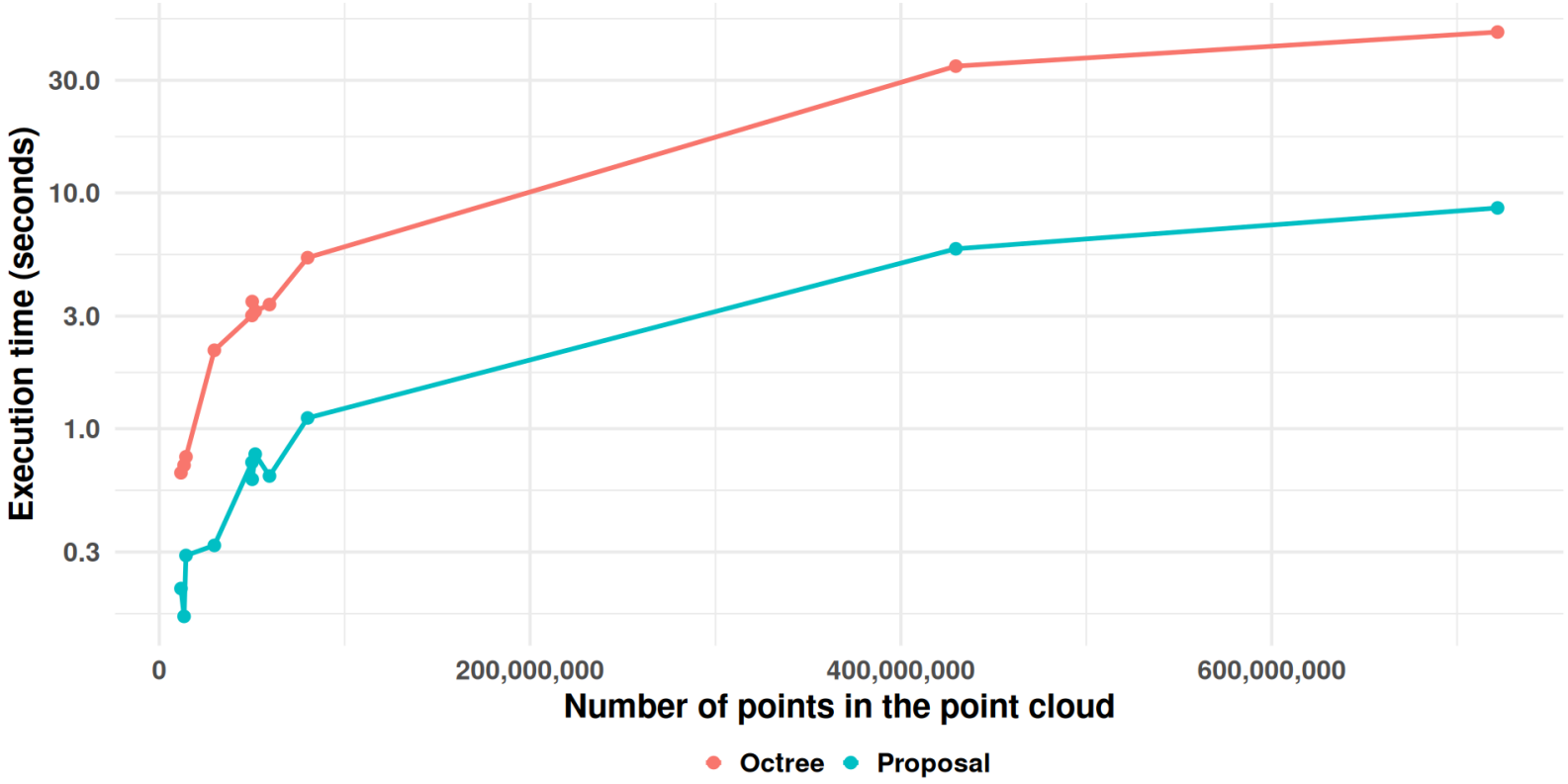
Clouds colored based on the **bandwidth** of the points.

Quantitative analysis



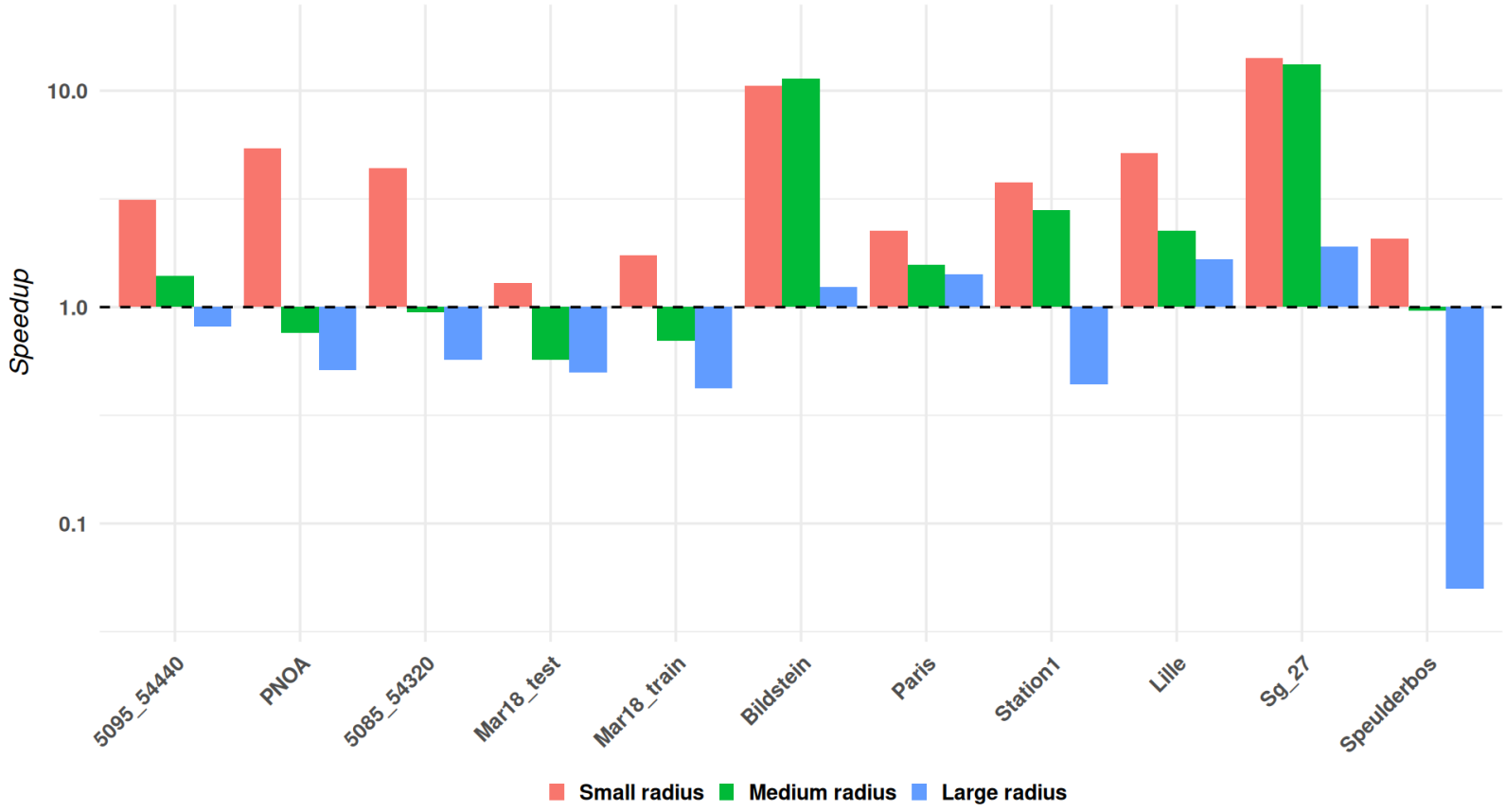
Average bandwidth reduction factor for point clouds.

Creation time



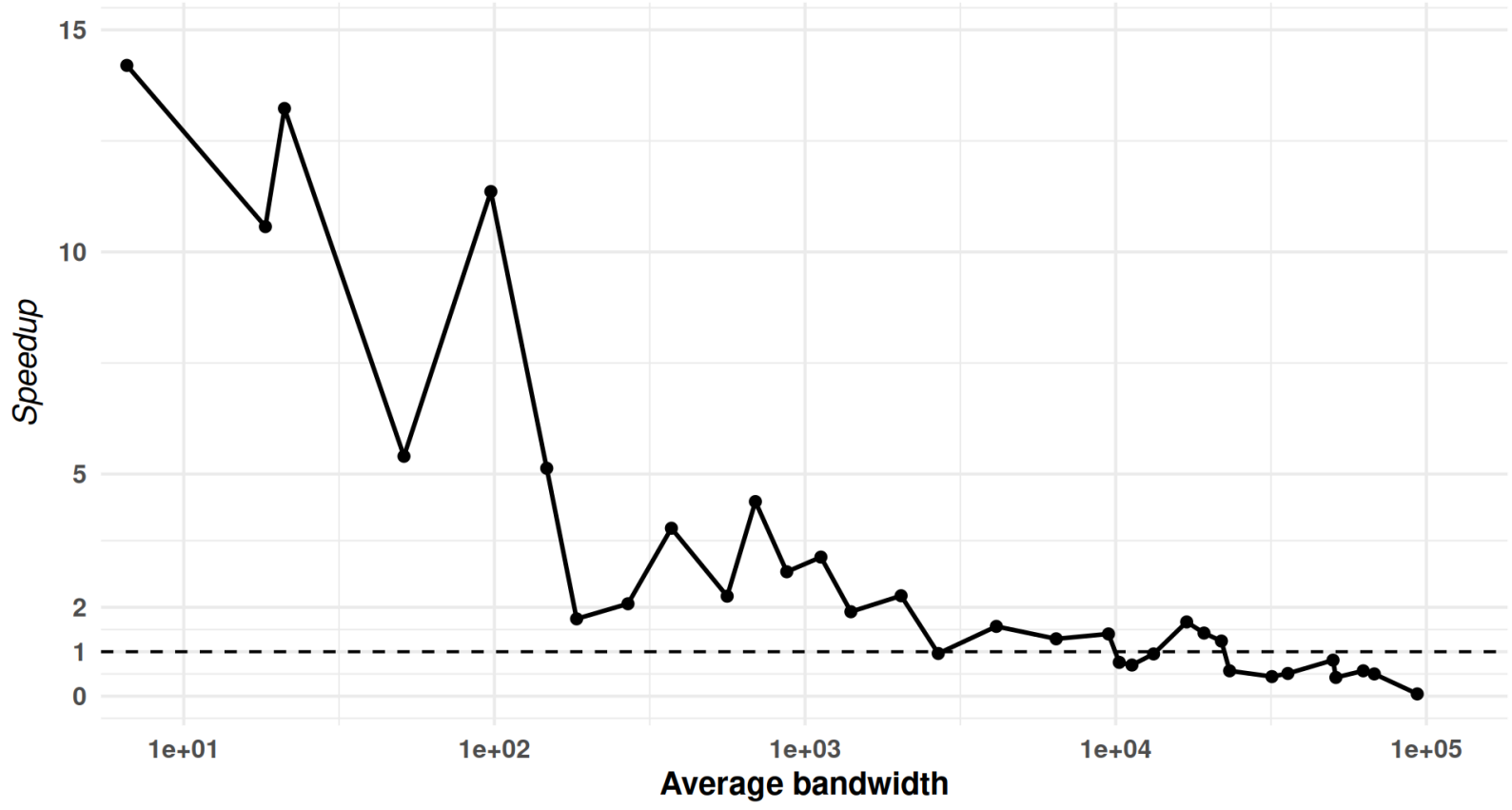
Runtime (seconds) of the octree structure and the proposed approach (logarithmic scale on the y-axis).

Runtime results



Speedup of the neighbors search (seconds) of the proposed approach with the octree.

Correlation between speedup and average bandwidth

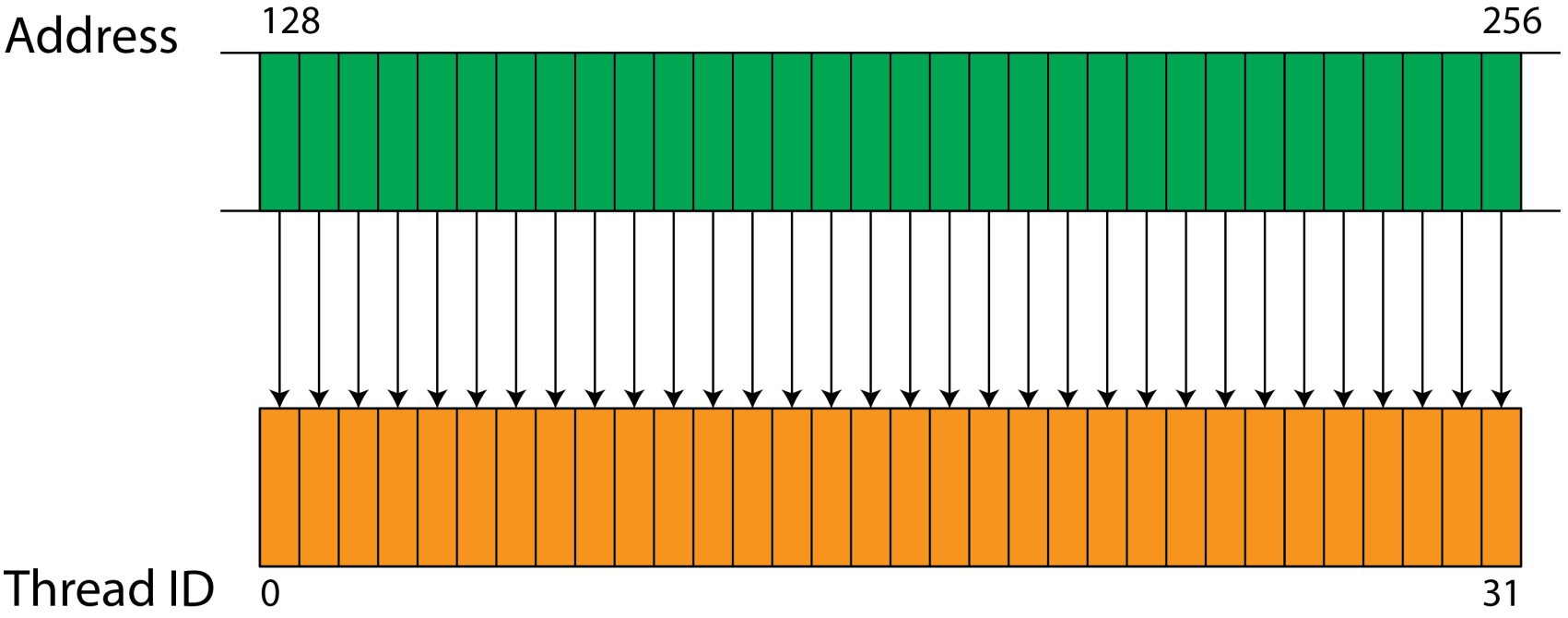


Conclusions

- Viable
- Improves performance
- Eliminates the need for complex auxiliary structures
- Opens up numerous lines of research
  - Use alternative ordering algorithms
  - Application to 3D point cloud segmentation
  - Explore an alternative architecture: GPU

Neighbor searching with GPU

Massive, **simple**, **independent** and **ordered** operations in memory.



Coalesced access to memory in GPU.

Matrix Ordering of 3D Point Clouds for Performance Improvement in Neighbor Searching Through Bounding

Abel Rodríguez Calleja: abel.rodriguez.calleja@rai.usc.es
Francisco Fernández Rivera: ff.rivera@usc.es
Miguel Yermo García: miguel.yermo@usc.es
Tomás García Barreiro: tomasgarcia.barreiro@usc.es



